VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

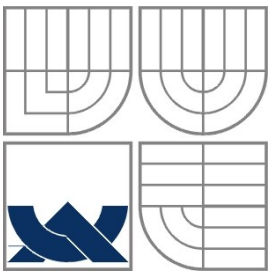
POROVNÁVÁNÍ DOKUMENTŮ NA ZÁKLADĚ BAREVNÉHO SCHÉMATU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

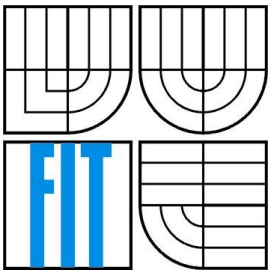
AUTOR PRÁCE
AUTHOR

MATÚŠ DUCHOŇ

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

POROVNÁVÁNÍ DOKUMENTŮ NA ZÁKLADĚ BAREVNÉHO SCHÉMATU

DOCUMENT COMPARISON BASED ON THE COLOR SCHEME

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MATÚŠ DUCHOŇ

VEDOUCÍ PRÁCE
SUPERVISOR

ING. MARTIN MILIČKA

BRNO 2012

Abstrakt

Tato práce pojednává o porovnávání webových dokumentů na základe jejich barevného schématu. Zaměřujeme se na problematiku získávání barevné schémy, pro ktorou byly navrženy dvě metody spracování dokumentu, obě využívající knihovnu CSSBox [1]. Dále je také představen návrh normalizace dokumentu, která získavání barevného schématu předchází. Její hlavním cílem je identifikovat a zjednotit možný promněnlivý obsah zkoumaného dokumentu, aby neovlivňoval výsledky nežádoucím spůsobem. Řeší se také implementace těchto navrhovaných postupů do knihovny, která byla v závěru práce testována na vzorové sade webových dokumentů.

Abstract

This paper proposes the method of web document comparison based on the color scheme. It focuses on issues of color scheme retrieving. There are suggested two methods that are using CSSBox library [1]. Also, a suggestion for a document normalization is introduced. Its main goal is to identify and unite possible dynamic content of given document which could negatively affect results of retrieving. Suggested procedures are implemented in the library. It was tested on a set of web documents.

Klíčová slova

Barevná schéma, porovnávání, dokumenty, web, historgam, CSSBox.

Keywords

Color scheme, comparison, documents, web, histogram, CSSBox.

Citace

Matúš Duchoň: Porovnávání dokumentů na základě barevného schématu, bakalářská práce, Brno, FIT VUT v Brně, 2012

Porovnávání dokumentů na základě barevného schématu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Martina Miličky. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Matúš Duchoň
16.5.2012

Poděkování

Chcem poďakovať všetkým, ktorí mi pri písaní tejto práce síce nedokázali pomôcť, ale ochotne si ma zakaždým vypočuli (čo väčšinou dokázalo aktuálny problém vyriešiť) a taktiež vedúcemu práce, za rady a správnu motiváciu.

Obsah

Úvod

Pri prezentácii súčasných webových dokumentov je autormi kladený dôraz na zaujímavú prezentáciu ich obsahu. Vizuálny vnem sa tak stáva podstatnou súčasťou toho, ako sa čitatelia v dokumente orientujú, ako je ovplyvnená ich schopnosť vnímania textu a hlavne ako si tento dokument zapamätajú. Naše vnímanie dokáže dokument najrýchlejšie rozoznať podľa farebnej kombinácie, ktorá je v ňom obsiahnutá. Preto je vizuálnym dojem zaujímavý aj z hľadiska vyhľadávania dokumentov, či v prípade phishingu, kedy útočník čitateľovi podvrhuje falošný dokument.

Cieľom tejto práce je preto navrhnúť a implementovať metódu, ktorá dokáže z daného webového dokumentu získať jeho farebnú schému a pomocou nej vyhodnotiť podobnosť s iným dokumentom.

Za týmto účelom je v nasledujúcej kapitole opísaný formát webového dokumentu a spôsobom akým, sú prostredníctvom kaskádových štýlov upravované jeho grafické vlastnosti. Ďalej sú zhrnuté jednotlivé prístupy k problematike porovnávania dokumentov a obrazov všeobecne a záver kapitoly je venovaný spôsobom reprezentácie farebnej schémy dokumentu prostredníctvom histogramov a možnosti ich porovnávania.

Na základe rozobratých poznatkov sú v druhej kapitole vypracované návrhy metód, ktoré budú získavať farebnú schému rozličným prístupom k analýze dokument. Rozobraté sú aj problémy, s ktorými sa pri ich analýze stretávame, napr. s premenlivý obsahom dokumentu. K nim sú vypracované možné riešenia. Taktiež je rozobratý konkrétny spôsob utvárania histogramu farebnej schémy vo vybranom farebnom priestore.

Výsledné riešenie je potom implementované v jazyku Java. Podrobnostiam implementácie sa venuje tretia kapitola. Na záver sú realizované metódy testované prostredníctvom porovnávania sady webových dokumentov, v ktorej sledujeme nakoľko sa približujú ľudskému hodnotenie ich podobnosti. Ich výsledky sú potom zhrnuté a na ich základe sa navrhujú možné rozšírenia. Popisy sady testovacích elementov a dokumentácia vytvorenej knižnice sú uvedené v prílohe práce.

1 Analýza

1.1 Webový dokumentu a jeho štruktúra

Pod pojmom webový dokument máme v našej práci na mysli dokument zapísaný v značkovacom jazyku HTML alebo XHTML, primárne určený na prezeranie vo webovom prehliadači. Prvú špecifikáciu jazyka HTML vypracoval na prelome 90tych rokov Tim Berners-Lee [2]. Jeho koncepcia vychádza z jazyka SGML. HTML bolo pôvodne navrhované na vytváranie a jednoduché zdieľanie dokumentov v prostredí siete Internet. Časom sa však táto koncepcia značne rozšírila. V dobe písania tejto práce je poslednou platnou špecifikáciou HTML vo verzii 4.01. V praxi je však už súčasnými modernými prehliadačmi podporovaná aj nová, doteraz oficiálne nevydaná verzia HTML 5.

Dokument v jazyku HTML je formovaný pomocou značiek (*tags*), ktoré utvárajú jednotlivé elementy. Každá z definovaných HTML značiek má v dokumente svoj štrukturálny resp. sémantický význam. Väčšina značiek je párnych – obsah elementu, ktorý značka vytvára, je ohraničený otváracou a uzatváracou značkou (napr. `<p>text odstavca</p>`). Používajú sa však aj nepárne značky ako napr. `<img/>`, ktorou je do dokumentu vložený obrázok, alebo `<br/>` pre zalomenie textu na nový riadok, etc.

Povinná štruktúra dokumentu pozostáva z koreňového elementu značky `<html>`, ktorý obsahuje hlavičku dokumentu pod značkou `<head>` a samotný obsah uložený v elemente značky `<body>`. V hlavičke dokumentu sú uvedené informácie ako metadáta (autor, kľúčové slová…), názov dokumentu, skripty a definície grafických vlastností elementov CSS (ktorými sa v tejto práci budeme ďalej zaoberať), resp. odkazy na externé súbory ktoré ich obsahujú. Obsah elementu `<body>` je už priamo zobrazovaný napr. po otvorení dokumentu v okne webového prehliadača. [3]

Poznatky o štruktúre a sémantike HTML dokumentu neskôr využívame v kapitole 2.1.

1.1.1 Document object model (DOM)

Zo zdrojového kódu webového dokumentu je pri jeho spracovaní (napr. vo webovom prehliadači) vytvorený objektový model dokumentu (DOM). Ten predstavuje aplikačné programovateľné rozhranie, ktoré vytvára model štruktúry daného dokumentu, umožňuje prechádzať ho, meniť a pristupovať k jeho jednotlivým vlastnostiam. Logická štruktúra DOM je typicky stromová. Každý dokument obsahuje najviac jeden uzol určujúci typ dokumentu a jeden koreňový uzol, ktorým je element `<html>`. Uzly sa rozdeľujú na elementy a na textové uzly. Kým elementy utvárajú samotnú štruktúru stromu (dokumentu), textové uzly sú listami tohoto stromu, predstavujúcimi textový obsah stránky. Textové uzly teda nemajú ďalších potomkov a nie sú ani nositeľmi informácie o svojom grafickom formátovaní v zobrazovanom dokumente. Tá sa uchováva v rodičovskom elemente.

1.1.2 Formátovanie pomocou kaskádových štýlov CSS

Pre potreby formátovania webových dokumentov, bol navrhnutý jazyk *Cascading Style Sheets* (CSS) [4]. Kaskádové štýly predstavujú jednoduchý mechanizmus pridávania grafických štýlov (napr. farby textu, pozadia, veľkostí písma…) elementom vo webových dokumentoch. Sú navrhnuté tak, aby bolo možné oddeliť obsah dokumentu (zapísanom v značkovacom jazyku) od jeho vizuálnej prezentácie. V súčastnosti poslednou platnou špecifikáciou je CSS level 2.1.

Štýly sú zoskupované do pravidiel. Každé pravidlo pozostáva zo selektoru (alebo viacerých selektorov) a bloku deklarácií. V ňom sa nachádza zoznam deklarácií – dvojíc grafická vlastnosť a jej hodnota hodnota. Selektor predstavuje predpis, podľa ktorého sú v dokumente vyhľadané elementy, na ktoré sa majú grafické vlastnosti zo zoznamu aplikovať. Obsahom predpisu môže byť názov elementu, alebo hľadaná hodnota jeho atribútu (špeciálne sa rozlišujú hodnoty atribútov `class` a `id`). K dispozícii sú aj tzv. pseudo-selektory, ktorými vieme priradiť predpis elementu v danom stave, napr. pri príchode kurzoru myši nad element, pre element, ktorý je prvým potomkom rodiča, etc. Je nimi tiež možné zacieliť na tzv. pseudo-elementy, napr. prvé písmeno textu, prvý riadom etc.

V kaskádových štýloch sa uplatňuje dedičnosť grafických vlastností textu. Tie sa z elementu ktorému boli pôvodne priradené prenášajú na potomkov. Dedičnosťou, alebo priradením viacerých pravidiel jednému elementu, vzniká potreba rozlišovať ktorá hodnota grafickej vlastnosti (deklarovanej vo viacerých pravidlách) sa nakoniec prejaví. Pravidlá sa preto do seba skladajú tzv. kaskádou. Tá spočíva vo vypočítaní váhy každej vlastnosti (podľa tvaru selektoru, poradia zápisu…) – vlastnosť s najvyššou váhou prevládne. Váhu vlastnosti je možné manuálne zvýšiť pridaním prívlastku `!important` do jej hodnoty. [5]

Pri tvorbe dokumentu môžu byť kaskádové štýly umiestnené buď priamo v ňom, ako obsah elementu `<style>`, alebo do externého súboru, na ktorý sa v dokumente odkazuje. Je tu ešte možnosť zadať ich aj priamo konkrétnemu elementu cez jeho atribút `style`. Takto zapísané grafické vlastnosti majú najvyššiu váhu.

Skupinu tried grafických štýlov je možné priradiť len pre zobrazovanie na určitom médiu, dajú sa tak upresniť grafika určená napr. pre zobrazení na monitore od grafiky ktorou je jeho obsah formátovaný pri tlači.

Stále aktuálnym problémom pri tvorbe webových dokumentov je plná podpora CSS vlastností zo strany webových prehliadačov. V súčastnosti sú už tvorcami webových prehliadačov aj autormi dokumentov podporované a využívané aj vlastnosti, ktoré prináša ešte stále len pripravovaná špecifikácia CSS level 3. Prehliadače však niektoré z nich podporujú len čiastočne, alebo si ich neoficiálnu špecifikáciu prispôsobujú (niektoré dokonca presadzujú svoje vlastné, ktoré špecifikácia nezahrňuje vôbec).

Aby mali autori webových dokumentov kontrolu nad rozdielnym spracovaním týchto grafických vlastností medzi prehliadačmi, väčšina prehliadačov ponúka možnosť zapísať vlastnosť aj s pridaným prefixom, ktorým sa rozlišuje pre ktorý prehliadač je táto vlastnosť (a jej hodnota) určená. To čiastočne rieši vzniknutý problém, no na druhej strane komplikuje prácu ako samotným autorom, tak aj “tretím stranám”.

Nástroj prezentovaný v našej práci sa síce zameriava na sledovanie grafický vlastností hlavne zo starších špecifikácií, ale nesprávna interpretácia niektorých z nich, alebo použitie novších grafických vlastností (napr. gradienty na pozadí, generované prehliadačom), sa môže podpísať pod nepresnosti v získanej farebnej schéme.

1.2 Porovnávanie dokumentov

Hromadná správa dokumentov so sebou často prináša potrebu vyhľadávania podobností resp. rozdielov medzi dokumentmi. S postupným pribúdaním rôznych druhov a formátov dokumentov sa menili aj nároky, ktoré sú na ich porovnávanie kladené. Postupne sa vyprofilovali tieto základné prístupy, ktorými je možné na problematiku porovnávania nazerať:

1.2.1 Porovnávanie textových dokumentov

Základnou a historicky najstaršou metódou porovnávania dokumentov je textové porovnávanie, ktoré sa zameriava na hľadanie podobností samotného textu. Sleduje zoskupenia znakov v riadku a usporiadania riadkov v dokumente. Tento druh porovnávania slúži hlavne pre vyhľadávanie zmien medzi rôznymi verziami zdrojových kódov/textov. Ako príklad nástroja, ktorý ho implementuje uvádzame program *diff*. Štruktúru ani vizuálne rysy dokumentu táto metóda nezohľadňuje.

1.2.2 Porovnávanie štruktúrovaných dokumentov

Pri hľadaní podobností medzi štruktúrovanými dokumentmi však už jednoduché textové porovnávanie neprináša požadované výsledky, pretože hierarchické usporiadanie dát v dokumente mu uniká. Preto je v tomto prípade potrebný prístup, ktorý dokáže rozpoznať štruktúru dokumentu a previesť ju do príslušného tvaru vhodného pre efektívne porovnávanie s inými dokumentmi.

Tento tvar sa označuje aj ako *signatúra* dokumentu. Tá sa snaží rozpoznávať a popisovať kľúčové znaky dokumentu a vzťahy medzi nimi.

V prípade štruktúrovaných dokumentov býva touto signatúrou najčastejšie graf, ktorý abstrahuje vzájomné usporiadanie jednotlivých prvkov dokumentu. Porovnávanie je potom realizované jednou z metód pre porovnávanie grafov. Tento spôsob porovnávania dokumentov je implementovaný napr. v [6].

Medzi štruktúrované sa jasne radia dokumenty zapísané v jazyku (X)HTML, rozobraté v predchádzajúcej podkapitole 1.1.

1.2.3 Porovnávanie dokumentov podľa vizuálu

Pre čitateľa prezerajúceho dokument je ale podstatná aj jeho vizuálna stránka. Ak teda potrebujeme vyhľadávať dokumenty na základe vizuálnej podobnosti, potrebujeme porovnávať signatúry získané z ich vizuálnych rysov.

Keďže webové dokumenty majú predpísaný základný formát zobrazovanie jednotlivých elementov, môžeme ich vizuálne rysy získať aj analýzou štruktúry, ako bolo ukázané v [7].

Ďalším prístupom je metóda získavania signatúry ako farebnej schémy dokumentu. Tá je potom reprezentovaná histogramom vytvoreným z farieb použitých v dokumente. Výsledná podobnosť dokumentov je potom získaná pomocou niektorej z metód na meranie vzájomnej vzdialenosti histogramov.

Okrem štruktúry môžme porovnávanie vizuálnej stránky riešiť tak, že ich najprv vykreslíme a ďalej sa zaoberáme porovnávaním ich výsledných obrazov (špeciálne potom ak je dokumentom samotný obrázok). Obraz dokumentov je takisto možné porovnávať prostredníctvom farebnej schémy. Jej výhoda oproti ostatným (ďalej uvádzaným) metódam porovnávania dokumentov spočíva v tom, že je odolná (*invariantná*) voči otočeniu obrazu a čiastočnej aj voči zmene jeho veľkosti. Taktiež je v porovnaní s ostatnými metódami rýchlejšia.

Farebné schémy, získané analýzou dokumentu, alebo až z jeho obrazu, možno reprezentovať a porovnávať pomocou histogramov, ako bude priblížené v 1.4. V našej práci sa zameriavame práve na riešenie tohto prístupu k porovnávaniu dokumentov vo webovom prostredí.

Vhodným porovnávaním farebných schém dokážeme uspokojivo porovnávať obrazy podľa farieb ktoré obsahujú, avšak nezohľadňujeme pri tom ich obsah a kompozíciu prvkov v obraze, napr. rozmiestnenie jednotlivých častí layoutu webového dokumentu. Na získavanie týchto informácií, potrebujeme použiť techniku segmentácie obrazu alebo techniku detekcie hrán v obraze. Segmentácia identifikuje kľúčové farebné plochy obrazu, na základe ktorých je potom možné realizovať porovnávanie. Detekcia hrán sa zameriava na hľadanie významných kontúr obrazu, signatúru potom predstavuje ako orientácia týchto hrán vzhľadom k okrajom dokumentu. Tieto metódy boli napr. použité použité pri porovnávaní obrázkov v [8] a porovnávaní webových dokumentov v [9].

Spomínané techniky ale nie sú invariantné voči transformáciám obrazu. Preto sa dnes pre robustnejšie porovnávanie podobností obrazov, založenom na obsahu, využíva hlavne metóda *SIFT* (*scale invariant feature transform*), ktorú popísal David G. Lowe v [10], alebo jej modernejšia varianta *SURF* (*speed-up robus features*) [11]. Tie v obraze vyhľadávajú kľúčové oblasti, ku ktorým sú vypočítané deskriptory (takzvaným. *features*). Získané deskriptory sú už odolné zmene veľkosti, natočeniu, rozmazaniu a zmene osvetlenia obrazu.

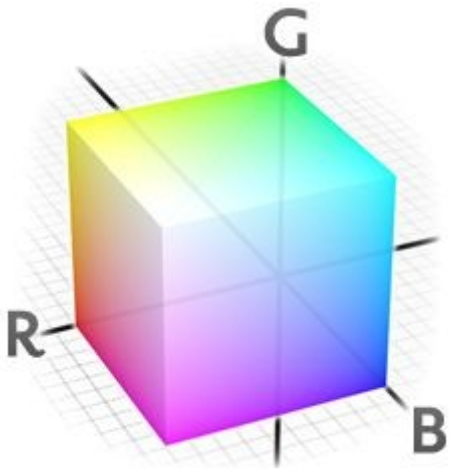
1.3 Reprezentácia farieb

Pre potreby zobrazovania farieb v počítačovej technike máme k dispozícii hneď niekoľko farebných modelov. Tie pre popis výslednej farby zvyčajne obsahujú jeden až štyri rozmery reprezentované buď farebnou zložkou alebo farebným kanálom. Pri porovnávaní obrázkov sú bežne používané nasledujúce farebné modely. Popis farebných modelov preberáme z [12].

1.3.1 Model RGB

RGB je aditívny model obsahujúci tri kanály základných farebných zložiek – červená (*Red*), zelená (*Green*) a modrá (*Blue*). Ako aditívny model vychádza z fyzickej povahy svetla, ktorou je skladanie lúčov rôznych vlnových dĺžok. Výsledná farba je vytvorená bodmi vyžarujúcimi základné farebné zložky (s rôznou intenzitou), ktoré sú umiestnené v tesnej blízkosti pri sebe alebo sa zobrazujú v rýchlom časovom slede. Farby dvoch a viacerých bodov sú potom vo výsledku vnímané ako jedna výsledná farba. Farebný priestor RGB je možné geometricky zakresliť ako kocku 1.1.

Tento model je presadzovaný doteraz používanou zobrazovacou technikou (OLED, LCD a CRT monitory), pretože jednotlivé pixely obrazu môžu byť bez ďalšieho prepočtu priamo mapované na fyzické bunky zobrazovacej plochy zariadenia.



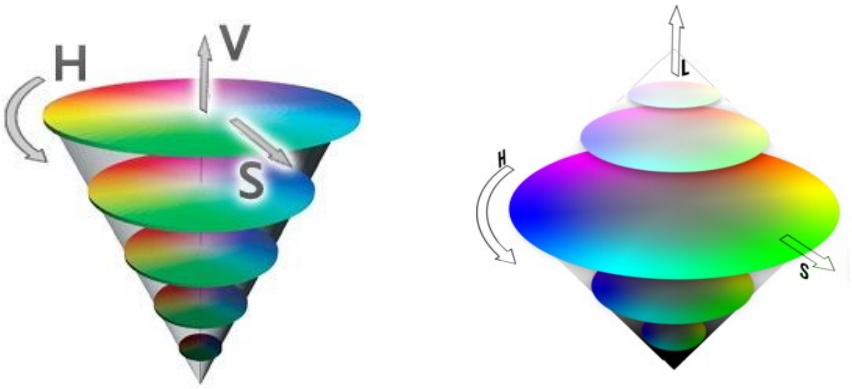
Obrázok 1.1: Reprezentácia farebného priestoru RGB, osy reprezentujú jednotlivé farebné zložky: červenú, zelenú a modrú. Prevzaté z [13].

1.3.2 Modely HSV, HSL

Táto skupina modelov sa snaží farebný priestor popísať tak, aby jednotlivé zložky lepšie zodpovedali atribútom, ktoré intuitívne vníma ľudské oko – odtieň farby, jej sýtosť a jas. HSV (často označované aj ako HSB) Tieto hodnoty priamo zaznamenáva v troch kanáloch (*Hue*, *Saturation*, *Value – Brightness*). Príbuzné modely poslednú zložku implementujú mierne odlišne, konkrétne v modeli HSL je ňou svetelnosť (*Lightness*) a v modely HSI intenzita farby (*Intensity*). Práve zmena poslednej zložky (jasu) najviac ovplyvňuje subjektívne vnímanú farbu.

Nevýhodou týchto modelov je nutnosť netriviálneho prepočtu z iného farebného modelu, najčastejšie z rozšíreného RGB, ktorá nie je vratná. Tá je však v porovnaní s konverziou medzi inými farebnými modelmi relatívne jednoduchá. Takisto *gamut* (rozsah plochy farebného priestoru, ktorú dokážeme pokryť daným modelom) nezachytáva celý priestor farieb, ktoré dokáže ľudské oko rozlišovať. To však neprevažuje výhody, ktoré nasadenie týchto modelov prináša pri vyhodnocovaní a hľadaní podobností v obrazoch, na čo sú často nasadzované.

Farebné modely HSV a HSL sa dajú v priestore znázorniť ako kužeľ, resp. dvojica podstavou spojených kužeľov 1.2.



Obrázok 1.2: Reprezentácia farebného priestoru HSV (vľavo) a HLS (vpravo). Prevzaté z [13]

1.4 Histogram

Farebnú schému dokumentu je možné reprezentovať pomocou farebného histogramu. Keďže farebné modely HSV a RGB, s ktorými sa v práci zaoberáme, sú definované viacerými kanálmi, farebná informácia je potom reprezentovaná buď ako trojica jednorozmerných histogramov, alebo ako jeden trojrozmerný histogram [14] (s ním budeme ďalej pracovať). Histogram predstavuje pravdepodobnosť výskytu konkrétnej farby v množine všetkých bodov (pixelov) sledovaného obrazu. Trojrozmerný histogram sa potom dá formálne zapísať nasledovne.

$$h_{A,B,C}(a,b,c) = N \,.\, Prob(A=a, B=b, C=c)$$

V tomto zápise predstavujú A, B a C jednotlivé farebné kanály a N je celkové množstvo pixelov, ktoré obrázok obsahuje. [15]

Keďže ľudské oko len veľmi slabo vníma farby nachádzajúce sa vo farebnom priestore blízko seba, je vhodné farby zaznačované do histogramu najprv kvantifikovať – zredukovať na menší počet relevantných farieb. Spojením susedných položiek sú vytvorené tzv. *biny*, do ktorých sa pôvodná farba redukuje. Pri vhodnom kvantifikovaní (pre farebný priestor v ktorom je histogram vytvorený) tak dokážeme podstatne zefektívniť napr. neskoršie porovnávanie histogramu bez toho aby to významne ovplyvnilo informácie o farebnej schéme, ktorú reprezentuje. Taktiež je vhodné pracovať s normalizovaným histogramom, v ktorom je zastúpenie jeho jednotlivých zložiek pomerné ku všetkým farebným zložkám obrazu.

1.4.1 Porovnávanie histogramov

Pre potreby porovnávania histogramov je k dispozícii viacero osvedčených techník. Podľa prístupu k problematike je ich možné rozlíšiť na vektorové a pravdepodobnostné metódy. Klasickou technikou porovnávania histogramov je napr. *Komolgorov-Smirnov test.* [16] Vo všeobecnosti sa však pravdepodobnostné metódy nepovažujú za vhodné na porovnávanie histogramov, reprezentujúcich

farebnú schému. Vizuálna podobnosť sa totiž lepšie chápaná ako príbuznosť farieb než blízkosť funkcie hustoty pravdepodobnosti. [15]

Rozšírenými metódami porovnávania farebných histogramov sú napr. Euklidovská metrika (*Euclidean distance*), Prienik histogramov (*histogram intersection*), Kosínova (*cosine distance*) alebo Kvadratická metrika (*quadratic cross distance*). Ďalej sa zameriame na prvé dve, ktoré bližšie popíšeme. Tie boli popísané a použité v [14] a [15], odkiaľ preberáme ich definície.

1.4.1.1 Euklidovská metrika

Nech ***h*** a ***g*** reprezentujú dva histogrami. Vzdialenosť medzi histogramom ***h*** a ***g*** môže byť pomocou Euklidovskej metriky vypočítaná takto:

$$d^2(h,g)=\sum_A \sum_B \sum_C (h(a,b,c)-g(a,b,c))^2$$

Pri tejto metóde sú porovnávané len navzájom príslušné položky histogramov s rovnakým počtom položiek.

1.4.1.2 Prienik histogramov

Prienik histogramov ***h*** a ***g*** je určený ako:

$$d(h,g)=\frac{\sum_A \sum_B \sum_C min(h(a,b,c),g(a,b,c))}{min(|h|,|g|)}$$

kde $|\boldsymbol{h}|$ a $|\boldsymbol{g}|$ sú magnitúdou histogramu, ktorá je daná počtom vzorkou ktoré histogram obsahuje. Výsledná suma je normalizovaná histogramom s menším počtom vzoriek.

2 Návrh riešenia

Cieľom navrhovaného riešenia má byť metóda, ktorou dokážeme posúdiť podobnosť dvoch webových dokumentov, na základe porovnávania ich farebnej schémy. Metóda má byť navrhnutá tak, aby čo najvernejšie zodpovedala ľudskému vnímaniu podobností vo farebnom priestore. Výslednú podobnosť treba vhodne reprezentovať.

Signatúru dokumentu, s ktorou bude naša metóda pracovať, predstavuje farebná schéma získaná z elementov, ktoré obsahuje DOM dokumentu.

Pri riešení treba uvažovať aj na možnosťou hromadného porovnávania dokumentov. V takom prípade je neefektívne opakovane znovu získavať signatúru dokumentu pred každým porovnávaním (okrem prípadu, že daný v danom dokumente došlo medzi časom k úpravám obsahu). Výhodnejšie je preto získanú signatúru upravovať do tvaru v ktorom môže byť uložená (napr. do databázy) a neskôr pri samotnom porovnávaní rýchlo dostupná a transformovaná do použiteľného tvaru.

Navrhovaný postup je nasledujúci:

1. Normalizácia obsahu dokumentu získaného dokumentu
2. Priradenie kaskádových štýlov k jednotlivým elementom v dokumente
3. Analýza spracovaného dokumentu niektorou z navrhovaných metód získavania farebnej schémy
4. Vytvorenie histogramu reprezentujúceho farebnú schému.
5. Porovnanie histogramov za pomoci implementovaných metód.

2.1 Normalizácia dokumentu

Predstavme si ale situáciu, kedy máme dva štruktúrou rovnaké dokumenty, ktorým je priradený i rovnaký súbor s grafickými štýlmi, avšak obsahové časti dokumentu sú u oboch rozdielne. Porovnávanie takýchto dokumentov by pre obe navrhované metódy mohlo dopadnúť s neočakávaným výsledkom, keďže rozdielny obsah spôsobí minimálne výraznejší nepomer medzi zastúpením jednotlivých farieb vo farebnej schéme.

Keďže sa zameriavame na porovnávanie podľa farebnej schémy, ktorá je v tomto prípade u oboch dokumentov identická. Očakáva sa teda, že výsledkom nášho porovnávania bude prehlásenie dokumentov za totožné (z hľadiska použitej farebnej schémy), alebo aspoň, že sa nachádzajú na hranici totožnosti. Dostávame sa tak k problému ako zohľadniť premenlivý obsah.

Uvažujme nad nasledujúcou vybranou časťou štruktúry dokumentu:

```
    <nav class=”navigation”>
  <a href=”/posts” class=”selected”>Latest posts</a>
    <a href=”/categories”>Categories</a>
    <a href=”/archive”>Archive</a>
  </nav>
</div>
```

```
<div class=”articles”>
   <article style=”color: darkblue, font-size: 14pt”>
      <h1>The commissioner of police</h1>
      <p>I am Stubb.</p>
      <p>But he said <a href=”/”>something else</a>.</p>
   </article>
   <article>
      <h1>Miss Collbran</h1>
      <p>The effect of this dispatch was instantaneous.</p>
      <p>At any rate I'll never go <i>there</i> again</p>
   </article>
   <article class=”featured-article”>
      <h1>My feet!</h1>
      <p>I think, this might be <i>it</i>.</p>
      <h2>When I was a boy</h2>
      <p>Or maybe, I don't.</p>
   </article>
</div>
<div class=”articles”>
   <ul class=”older-articles>
      <li><a href=”/posts/my-firts-day”>My first day</a></li>
      <li><a href=”/posts/do-it”>Do it!</a></li>
   </ul>
</div>
```

Pri bližšom pohľade na hore uvedenú štruktúru, si môžme všimnúť opakujúce sa sesterské elementy rovnakej značky. Opakovaní niektorých značiek medzi priamymi potomkami elementu nám napovedá, že jeho obsah môže byť premenlivý – pridaním ďalších elementov tejto značky sa už bude meniť len obsah, nie štruktúra dokumentu. Špeciálne to platí pri týchto HTML značkách s daným sémantickým významom:

- `<h1>`, `<h2>`, `<h3>`, … – nadpisy článkov
- `<p>` – odstavec textu
- `<a>` – odkazy, v texte alebo ako položky navigačného menu
- `<li>`, `<dt>`, `<dt>` – položky zoznamov

A pri značkách ktoré sa objavujú v pripravovanej verzii HTML 5:

- `<article>` – element ohraničujúci článok
- `<section>` – element ohraničujúci kapitolu
- `<nav>` – navigačné menu

Predpokladajme teda, že ak v štruktúre dokumentu narazíme na niektorú z vymenovaných značiek, ktorá sa v jednej úrovni viackrát opakuje, môžeme ich identifikovať ako premenlivý obsah.

Ďalším krokom bude tieto elementy zredukovať do takej podoby, aby sa spomínané dokumenty stali identické aj po obsahovej stránke, ale zároveň aby boli ich grafické vlastnosti zachované v čo najväčšej miere a zohľadnené vo výslednej farebnej schéme.

Obsah je redukovaný pravidlom, podľa ktorého môže byť medzi priamymi potomkami elementu každá značka zo spomínanej skupiny obsiahnutá iba raz. Ponechávame si tak len prvého potomka s danou značkou a ďalších potomkov s rovnakou značkou zahadzujeme.

Zredukovaný element môže obsahovať aj grafické pravidlá, ktoré sú mu pridelené atribútom `style`. Tieto pravidlá však chápeme len ako lokálne zvýraznenie konkrétneho elementu a preto ich pri normalizácii obsahu zahadzujeme aby neovplyvňovali získanú farebnú schému dokumentu.

Častou praktikou pri tvorbe dokumentov ale býva zvýraznenie niektorých elementov, ktorých vplyv na farebnú schému je žiadaný, no po aplikovaní uvedených pravidiel by mohli byť tieto elementy z dokumentu vypustené. Stáva sa tak napr. pri zvýraznení aktuálnej položky v navigačnom menu, zvýraznení prvého odstavca v článku, etc. Preto je naše pravidlo ešte potrebné upraviť tak, že podmienkou pre vyradenie potomka bude okrem zhody v značke, aj zhoda v triede, ktorú má priradenú atribútom `class`.

Každému elementu môže byť atribútom `id` priradený jedinečný identifikátor v rámci dokumentu. Väčšinou sa jedná o elementy, ktoré v štruktúre stránky zohrávajú významnú úlohu. Taktiež im ich grafické vlastnosti bývajú často adresované CSS selektorom, ktorý zameriava práve atribút id.

Toto pravidlo je ešte nutné prispôsobiť pre obsah niektorých elementov, v ktorých má nielen redukovať opakované značky, ale taktiež zabezpečiť aby obsahovali značky, ktorých grafické vlastnosti bývajú autormi webových dokumentov často prispôsobované a ovplyvňujú tak výslednú farebnú schému, aj keď nemusia byť prítomné v každom dokumente.

Prvou takouto značkou je `<p>` ohraničujúca odstavec textu článku. Skúsenosť ukazuje, že text v odstavci veľmi často obsahuje odkazy vyznačené značkou `<a>` a takisto aj zvýraznenia textu pomocou značiek `<b>` a `<i>`. Ich prítomnosť v odstavci však nemáme nijak zaručenú. Preto sa ich najprv pokúsime redukovať vyššie popísaným pravidlom, a v prípade že chýbajú, ich dopĺňame, aby sme sa ubezpečili že budú do získavanej farebnej schémy zahrnuté.

Ďalšou prípadom sú zoznamy, ktoré sa v štruktúre webových dokumentov častou používajú pre formátovanie navigácie. Jej vizuál sa podstatnou mierou podpisuje pod výsledný vizuálny dojem stránky a preto je zoznamom podobným spôsobom ako pri odstavcoch vynútený odkaz zapísaný značkou `<a>`.

Poslednú výnimku predstavuje element `<article>`. Súčasťou jeho obsahu bývajú spravidla aj značky nadpisov. Zabezpečujeme preto prítomnosť nadpisy `<h1>`, `<h2>` a `<h3>`. Nadpisy nižších úrovní sa už zvyčajne od ostatného textu po grafickej stránke výrazne neodlišujú a preto sa nimi nezaoberáme. Pri nadpisoch prvej úrovne `<h1>` je ešte možnosť, že sú ukryté v elemente značky `<header>`, ktorý označuje hlavičku článku, ktorej súčasťou býva aj nadpis prvej úrovne.

Po aplikovaní navrhovaných pravidiel normalizácie obsahu, by sa mala štruktúra časti dokumentu upraviť nasledovne :

```
    <nav class=”navigation”>
  <a href=”/posts” class=”selected”>Latest posts</a>
```

```
    <a href="/categories">Categories</a>
</nav>
<div class="articles">
    <article>
        <h1>The commissioner of police</h1>
        <h2></h2>
        <p>I am Stubb.<a></a><b></b><i></i></p>
    </article>
    <article class="featured-article">
        <h1>My feet!</h1>
        <p>I think, this might be <i>it</i>.<a></a><b></b></p>
        <h2>When I was a boy</h2>
    </article>
</div>
```

Pri normalizácii sa nezaoberáme značkami `<div>`. V sémantike HTML síce nemajú priradený žiaden význam [3], používané sú ale hlavne pre tvorbu štruktúry layoutu dokumentu. Mohli by sme sa síce spoliehať na navrhované pravidlo, ktoré redukuje elementy len pri zhode v značke, triede a identifikátore, ale takto nie je zaručené, že redukovaný element `<div>` neobsahuje vizuálne odlišný obsah, ktorý môže zásadne ovplyvniť získanú farebnú schému a pri redukcii by bol zahodený.

Podobne bez sémantického významu sú aj elementy značky `<span>`. Tá sa zasa zvykne využívať pre lokálne zvýrazňovanie textu (väčšinou pomocou atribútu `style`), obzvlášť v elementoch odstavca `<p>` (a článku `<article>`). Lokálne zvýraznenia v texte by mohli nežiadúco ovplyvniť výslednú farebnú schému. Preto je element značky `<span>` spomedzi potomkov elementov `<p>` (a `<article>`) zahadzovaný. A keďže túto časť štruktúry stránky už chápeme ako samotný obsah, v ktorom si neželáme zvýraznenia, odoberáme aj ich zvyšným potomkom prípadný atribút `style`.

Navrhovaným postupom by sa nám malo podariť nie len normalizovať obsah dokumentu ale, takisto zabezpečiť aby boli v texte dokumentu obsiahnuté zvýrazňovacie značky, ktoré prinesú detailnejšie informácie o použitej farebnej schéme.

2.1.1 Obrázky a farebná schéma

O obrázkoch v dokumente uvažujeme výhradne ako o premenlivom obsahu. Pri normalizácii dokumentu sú preto obrázky zámerne odstraňované. Výnimkou sú však obrázky na pozadí elementov, ktoré sú definované grafickou vlastnosťou `background-image`.

Prvá nami navrhovaná metóda – metóda prechádzania objektového modelu dokumentu – však nezohľadňuje ani obrázky na pozadí. Keďže nesleduje ani priestorové rozmiestnenie elementov, nedokážeme pri nej ani vyhodnotiť akým spôsobom je obrázok pozadia umiestnený relatívne k ostatným elementom a aký vplyv by teda malo zahrnutie farebnej schémy obrázka do získavanej farebnej schémy dokumentu. Spoliehame sa preto len na dobrý zvyk tvorcov webového dokumentu, ktorým býva spolu s obrázkom pozadia definovať aj vhodnú substitučnú farbu, ktorá dokáže obrázok nahradiť, napr. v prípade že sa ho nepodarí načítať.

Keďže druhá metóda vytvára na základe získaných grafických vlastností obraz dokumentu, mala by byť schopná informácie o obrázku na pozadí interpretovať a mi ich následne vieme zahrnúť do výslednej farebnej schémy.

Otáznym však môže byť prístup k obrázku, ktorý reprezentuje logo v hlavičke dokumentu. Logo spoločnosti na hlavičkovom papieri totiž môžeme chápať ako súčasť farebnej schémy. Keďže však HTML špecifikácia nedefinuje sémantiku vkladania loga do dokumentu, vývojári majú rozdielne prístupy k riešeniu tohto problému a logá sa tak v hlavičke môžu vyskytovať aj ako obrázky na pozadí elementu, tak aj ako obrázky vložené pomocou elementu `<img>`. Pri návrhu predstavovaného riešenia tak bol nakoniec presadený názor, že na logo treba nazerať len ako na obsahový prvok a preto by sa sme ho pri analýze dokumentu nemali zohľadňovať.

2.2 Spracovanie dokumentu a aplikácia grafických štýlov

V tomto procese bude treba správne priradiť jednotlivé grafické vlastnosti definované rôznymi triedami na príslušné elementy, pričom treba zohľadňovať pravidlá dedičnosti a kaskádového skladania štýlov. Taktiež je potrebné zabezpečiť základné formátovanie jednotlivých elementov, podobne ako sa o to starajú webové prehliadače.

Naše navrhované riešenie si vystačí s podporou grafických vlastností podľa špecifikácie CSS 2.1, ktorých podpora v moderných prehliadačoch nie je až taká kritická.

2.3 Metódy získavania farebnej schémy

Na účely získavania farebnej schémy analyzovaného dokumentu, sú navrhované dva metódy, ktorých princíp si priblížime v nasledujúcich podkapitolách:

2.3.1 Metóda analýzy modelu dokumentu

Táto metóda vychádza z jednoduchej myšlienky postupného prechádzania štruktúrou dokumentu, pri ktorom sa do farebnej schémy zaznamenávajú grafické štýly jednotlivých elementov. Keďže pri tom nemáme spracované informácie o rozmiestnení a rozmeroch prechádzaných elementov, volíme možnosť získané grafické vlastnosti váhovať podľa ich významu, rovnako tak aj sémantického významu HTML značky elementu, ktorá tieto vlastnosti nesie. Napr. farba pozadia má väčší vplyv na celkový vizuálny dojem než farba textu a podobne väčší vplyv má nadpis oproti odstavcu.

2.3.2 Metóda analýzy výsledného obrazu dokumentu

Ak však chceme presnejšie zachytiť celkový dojem, ktorým farebná schéma zobrazovaného dokumentu pôsobí na čitateľov, musíme zohľadniť priestorové vlastnosti jeho elementov. Sme pri tom postavený pred náročnejšiu úlohu, ktorú predstavuje vytvorenie aspoň približného obrazu analyzovaného dokumentu. V ňom musíme správne interpretovať všetky zadané grafické vlastnosti elementov. Z vytvoreného obrazu je následne získaná farebná schéma dokumentu.

Výhodou tohto riešenia by bolo, že dokáže zohľadňovať aj vplyv obrázkov na pozadí, ktoré sa v súčasnosti vo webovej grafike využívajú v hojnej miere a farebnú schému dokumentu často ovplyvňujú zásadným spôsobom .

2.4 Farebná schéma dokumentu

Pri porovnávaní farebných schém plánujeme využiť trojrozmerný histogram, pracujúci s farebným modelom HSV, do ktorého je schéma transformovaná.

S vytváraním histogramu sú spojené dve faktory súvisiace s efektívnosťou a výkonom: schopnosť ľudského oka rozlišovať medzi príbuznými farbami a výpočtová náročnosť porovnávania histogramov obsahujúcich zložky všetkých farieb v danej farebnej schéme (pričom väčšina z nich nemusí byť v zdrojovom obraze prítomná). Výsledné nároky na procesorový čas by mohli ohroziť nasadenie vytvoreného nástroja hľadajúceho podobnosť signatúry zadaného dokumentu so signatúrami dokumentov uložených v databáze. Preto je vhodné každý kanál cieľového farebného priestoru diskretizovať na menší počet položiek a až tie následne zaznamenávať v histograme. Pri optimálnom množstve položiek histogramu, bude porovnávanie dosahovať najlepšie časy a pritom zachovávať čo možno najvernejšiu reprezentáciu farebnej schémy dokumentu. Pri vytváraní histogramu, obsahujúceho farby popísané farebným modelom HSV, môžeme vychádzať aj zo skutočnosti, že ľudské oko je pri rozlišovaní najcitlivejšie na zmenu svetlosti.

Keďže v dokumente sa môže vyskytovať relatívne veľké množstvo farieb, pričom niektoré sú vo výsledku zastúpené len v zanedbateľnom pomere, je vhodné nasadiť prah výskytu a farby, ktoré ho nepresahujú, do výslednej farebnej schémy nezahrňovať.

Pre porovnávanie použijeme dvojicu metód priblížených v kapitole 1.4.1: Euklidovskú metriku a metódu prieniku histogramov. Dvojica metód nám pomôže aj pri lepšom hodnotení dosiahnutých výsledkov porovnávania.

3 Implementácia

Výsledné riešenie je implementované v programovacom jazyku Java ako knižnica, obsahujúca nástroje pre načítanie, získanie a porovnanie farebnej schémy dokumentu viacerými metódami.

Java je objektovo orientovaný programovací jazyk. Jeho syntax jazyka vychádza z jazyka C a C++, Dôraz je pri tom kladený na prehľadnosť zdrojového kódu. Vývoj Javy začal už v prvej polovici 90tych rokov a aj keď bola pôvodne určená pre nasadenie v domácich spotrebičoch a multimediálnych zariadeniach, dnes nachádza uplatnenie v širokom spektre zariadení od mobilných telefónov až po serverové stanice. Okrem dobrej čitateľnosť prináša Java pomocou nástroja *garbage collector* vlastnú automatizovanú správu pamäte. Programátorovi odpadá úloha riešiť ju individuálne a tak sa znižuje priestor pre chyby, ktoré by mohli vzniknúť. Napísaný program je prekladaný do tzv, *bytecode*. Ten je potom spúšťaný v Java Virtual Machine (JVM), v ktorom je mapovaný na aktuálnu architektúru. Takto je zabezpečená prenositeľnosť kódu medzi rôznymi zariadeniam (pre ktoré je dostupná verzia JVM).

V nasledujúcich častiach tejto kapitoly budú spomenuté niektoré metódy a vzťahy medzi triedami vytvorenej knižnice. Podrobnosti o jej fungovaní a kompletnú dokumentáciu je možné nájsť v prílohe 2.

3.1 DOM parser

Úlohou parseru je spracovať zdrojový kód stránky a vygenerovať z neho objektovo orientovaný model dokumentu (DOM).

V našom riešení využívame na tento účel knižnicu *NekoHTML* [17]. Jeho výstupom je objektový model (X)HTML dokumentu implementujúci rozhranie Document z balíka org.w3c.dom [18]. To poskytuje metódy umožňujúce prechádzanie uzlami získaného dokumentu a manipuláciu s nimi – keďže elementy, textové uzly ani komentáre nemôžu existovať mimo dokumentu, poskytuje metódy, ktorými sa tieto uzly dajú vytvoriť a zaradiť do dokumentu).

Bohužiaľ, nie všetky publikované dokumenty, napísané v jazyku (X)HTML, dodržiavajú predpísané štandardy tohto jazyka. Preto aj keď NekoHTML dokáže spracovať i nevalidné dokumenty, nemáme zaručené, že bude výsledok zodpovedať spracovaniu tohto dokumentu vo webovom prehliadači. Parseri, ktoré využívajú ktoré využívajú, pristupujú k nevalidným zdrojovým kódom individuálne.

3.2 Rozsah plochy pokrytej textom

Pri získavaní farebnej schémy z normalizovaného dokumentu, ktoré bude rozobraté v nasledujúcom odseku, sa potrebujeme oprieť aj o údaj, približne koľko z plochy zabranej textom je reálne pokrytej jeho farbou.

Preto sme pripravili jednoduchý experiment, v ktorom sme sledovali ako na tento údaj vplýva zmena veľkosti písma. Vybrali sme 1000 znakový úryvok knižného textu, ktorý boli zobrazovaný vo

webovom prehliadači pri štandardnej výške riadku a rozostupe medzi znakmi. Z vykresleného dokumentu bola vybraná plocha ohraničujúca odstavec, obsahujúci náš text. V tomto obraze, prevedenom prahovaním do B&W, bol potom sledovaný pomer medzi vyfarbenými a nevyfarbenými pixelmi. Výsledky zahrnuté v tabuľke 3.1.

Rozmer písma	**10px**	**12px**	**14px**	**16px**	**18px**	**22px**	**28px**	**36px**	**Priemer**
Reálne pokrytá plocha	15,5%	15,7%	18,3%	18.3%	17,5%	18,8%	17,6%	17,4%	**17,3%**

Tabuľka 3.1: Reálna plocha pokrytá farbou textu na ploche odstavca

3.3 Získavanie farebnej schémy

Prvým krokom v procese získavania farebnej schémy dokumentu je normalizácia jeho objektového modelu podľa pravidiel navrhovaných v 2.1.

Stromovú štruktúru objektu dokumentu prechádzame po úrovniach metódou *breadth-first*. Zameriavame sa pri tom na uzly typu element. Ten spracovávame tak, že prechádzame jeho priamych potomkov. Znovu sa zameriavame len na uzly typu element a kontrolujeme či značka tohto elementu patrí do skupiny, ktoré redukujeme. Tieto uzly sú postupne ukladané do asociatívneho poľa, v ktorom je kľúčom kombinácia jeho značky a prípadnej triedy CSS, určenej atribútom `class` alebo identifikátorom elementu zadaným atribútom `id`. Ak už sa v tomto poli nachádza záznam s rovnaký kľúčom, vyhodnocovaný uzol odstraňujeme zo štruktúry dokumentu (spolu so všetkými jeho potomkami). Po normalizovaní priamych potomkov spracovávaného uzlu ešte skontrolujeme, či má daná značka predpísaný nejaký povinný obsah. Ten prípadne doplníme. Možnou slabou stránkou tohto prístupu je spoliehanie sa na správne využívanie sémantických značiek v danom dokumente.

Všetky spôsoby získavania farebnej schémy sú implementované ako samostatné triedy, ktoré rozširujú abstraktnú triedu `Signature`. V nej je okrem iného definovaná abstraktná metóda `retrieveSignature()`, ktorá je v dediacich triedach preťažená metódou obsahujúcou vlastný postup získavania farebnej schémy. Tento návrh robí knižnicu jednoducho rozšíriteľnú o ďalšie metódy získavania farebnej schémy z dokumentu.

Farebná schéma sa uschováva objekte triedy `ColorScheme`. Ten obsahuje metódu `addColor()`, ktorou sa do schémy pridáva záznam o farbe. Spolu s ním je možné zadať aj "množstvo" výskytu danej farby. Pri pridaní už existujúcej farby sa jej množstvo zaznamenané množstvo inkrementuje o zadanú hodnotu. Z celkové množstva farby je potom pri vyžiadaní zoznamu farieb vypočítavané pomerné zastúpenie jednotlivých farieb. Tento údaj slúži hneď pri zaradzovaní farieb do výsledného zoznamu – zaradené sú do neho len farby ktoré prekročili prahovú hranicu výskytu 0,05%. Farby s nižším konečným zastúpením vo farebnej schéme vyhodnocujeme len ako šum. Farby sa do farebnej schémy pridávajú a uchovávajú v modeli RGB. Získať ich môžeme volaním metódy `getColorsList()`, ktorá vráti zoznam štruktúr obsahujúcich farbu a jej zastúpenie v schéme.

Trieda `ColorScheme` obsahuje priamo metódu na vytvorenie histogramu zo získaných farieb, ktorý sa neskôr využíva pri porovnávaní farebných schém.

Ďalším krokom, po normalizácii získaného dokumentu, je analýza, pri ktorej sú na príslušné elementy aplikované všetkých grafických štýlov CSS, definovaných buď priamo v dokumente, v atribútoch style, alebo v nalinkovaných externých súboroch.

Na tento účel využívame CSS parser *JStyleParser* [19] ako súčasť knižnice CSSBox. Spracováva grafické štýly CSS 2.1 do štruktúry, z ktorej sa dajú efektívne priradiť cieľovým DOM elementom. Zahrnuté sú všetky grafické štýly, ktoré sú určené na zobrazovanie dokumentu vo webovom prehliadači.

Po tomto kroku sa už ďalší postup napĺňania farebnej schémy líši podľa použitej metódy porovnávania. Posledným spoločným krokom je že pri porovnávaní oboma metódami ignorujeme elementy značiek `<hr>`, `<script>`, `<style>`, `<img>`, `<audio>`, `<video>`, `<iframe>` a `<object>`.

3.3.1 Metóda analýzy modelu dokumentu (*DOMSignature*)

Metóda prechádza stromom modelu normalizovaného dokumentu metódou *breadth-first*. Každý uzol je spracovávaný tak, že sa k nemu priradí príslušný zoznam štýlov získaných analýzou dokumentu pomocou nástroja JStyleParser. Z neho si vyberáme vlastnosť `color`, ktorá definuje farbu textu a vlastnosť `background-color`, ktorou je určená farba pozadia. Takto získanú farbu vkladáme do farebnej schémy.

Chýbajúce priestorové informácie o veľkostiach a rozmiestnení elementu nahradzujeme váhovaním významu získanej farby elementu vo farebnej schéme. Využívame pri tom možnosť určiť počet jednotiek nájdenej farby pri pridávaní do objektu triedy `ColorScheme`, uchovávajúceho farebnú. Váhovanie zohľadňuje či sa jedná o farbu textu alebo pozadia

Koeficienty váhovania a základné počty jednotiek boli určené empiricky tak, aby zohľadňovali rozdiel medzi plochou ktorú vyplní farba pozadia a plochou ktorú pokrýva text a takisto sémantický význam niektorých značiek (napr. nadpisov).

V prípade ak spracovávaný element nemá učenú vlastnú farbu pozadia, snažíme sa ju umelo zdediť tak, že hľadáme prvého predka, ktorý ju definovanú má. Pri tom ale výrazne znížime váhu takto pridelenej farby. Robíme tak z toho dôvodu, aby veľké množstvo malých elementov bez definovaného pozadia neskresľovalo výslednú farebnú schému.

3.3.2 Metóda analýzy výsledného obrazu dokumentu (*ViewportSignature*)

Pri získavaní farebnej schémy touto metódou je potrebná ešte jedna normalizácia. Chystáme sa totiž zobrazovať aj textové uzly, ktoré ostali pri normalizácii dokumentu nepozmenené, pričom dĺžka ich obsahu sa môže výrazne odlišovať. Je preto potrebné ich analogicky normalizovať.

Model dokumentu je teda znovu prechádzaný metódou *breadth-first* a vyhľadávajú sa uzly typu `textNode`. Keďže text nepredstavuje homogénnu plochu, rozhodli sme sa ho nahradzovať blokovým elementom, ktorý kopíruje grafické vlastnosti textu – podľa farby textu definovanou vlastnosťou `color` je tomuto zastupujúcemu elementu nastavená farba pozadia `background-`

`color` a jeho výška je farba pozadia je mu nastavovaná na farbu textu zadanú vlastnosťou `color` a jeho výška zodpovedá vlastnosti `font-size`.

Zastupujúci element je blokový (vlastnosť `display: block`) a tak vypĺňa celú šírku riadku v rodičovskom elemente, podobne ako text ktorý nahradzuje. Ak zohľadníme údaje o ploche ktoré farba textu reálne pokrýva, získané v 3.2, vo farebnej schéme sa prejaví podobne ako približne 5 riadkov textu.

Výnimku tvorí text umiestnený v elementoch sémantickej značky určenej pre zvýrazňovanie obsahu – `<a>`, `<b>`, `<i>`. Keďže zvýrazneného textu býva v typickom odstavci oproti normálnemu výrazne menej, je výška zastupujúceho elementu pri zvýraznenom texte nastavená tak, tak aby zodpovedala výške jedného riadku v pôvodnom odstavci. Za každým zastupujúcim elementom je ešte, pomocou vlastnosti `margin-bottom`, pridaná malá medzera aby text v elemente, ktorý nemusí mať nastavené odsadenie, nezakrýval úplne prípadnú farbu/obrázok pozadia.

Pri tomto priechode štruktúrou dokumentu sa zároveň uistíme, že všetky elementy, so značkou ktorá slúži na formátovanie textu, nejaký aj obsahujú.

Po vykonaní týchto zmien však musíme znovu analyzovať dokument pomocou JStyleParser, aby sa prejavili uskutočnené zmeny a získali sme aktuálne mapovanie grafických štýlov k elementom dokumentu.

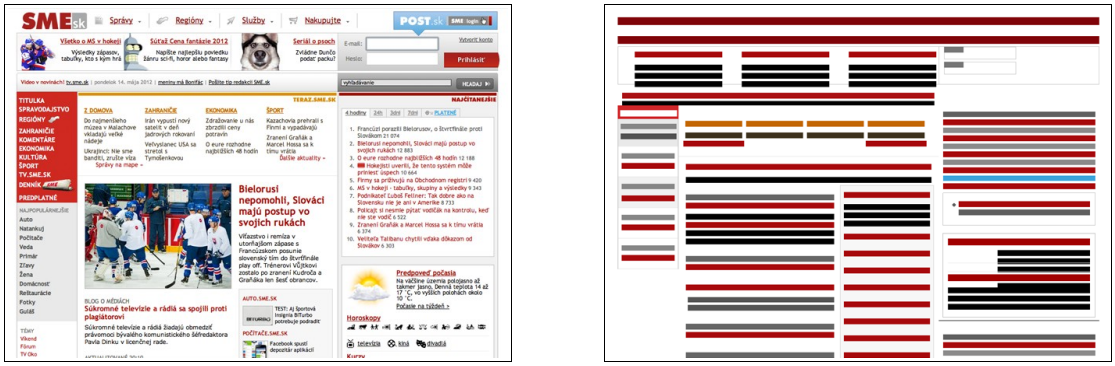
Takto spracovaný model dokumentu je už pripravený na vykreslenie jeho obrazu, ktorý v tejto metóde slúži ako podklad pri získavaní farebnej schémy. Na vytvorenie obrazu je použitá knižnica *CSSBox* [1]. CSSBox je nástroj na vykresľovanie (X)HTML dokumentov spolu s aplikovanými grafickým štýlmi CSS, implementovaný v jazyku Java. Jeho hlavným účelom je získavať kompletné informácie o zobrazovanej stránke a jej obsahu (implementované grafické vlastnosti jednotlivých elementov, ich rozmiestnenie a rozmery, zalamovanie textu…), ktoré je možné ďalej použiť. Na základe týchto informácií však umožňuje dokument aj sám vykresliť.

Vstupom pre CSSBox je koreň dokumentu z jeho objektového modelu, zoznam štýlov, získaných z tohto dokumentu, pomocou nástroja JStyleParser a rozmery okna, do ktorého má byť dokument premietnutý. Po spracovaní týchto parametrov získavame objektovo orientovaný model výstupu stránky, ktorý je možné priamo zobraziť, no je vhodný aj na ďalšie spracovanie a analýzu.

Nepísaným štandardom pri vytváraní súčasných webových dokumentov je rozmer layoutu 960px. Oknu, do ktorého bude dokument vykreslený, preto nastavujeme klasickú šírku obrazovky 1024px. Dávame tak tak priestor pre prejavenie grafických vlastností pozadia dokumentu. Výška výstupného obrazu je orezaná tak, aby zodpovedala výške elementu body. Predchádza sa tak nadmernému vplyvu pozadia na farebnú schému.

V pôvodnom návrhu tejto metódy bolo zamýšľané získavať farebnú schému výhradne z takto získaných informácií o elementoch. Ukázalo sa ale, že z vykresleného obrazu dokumentu (po jeho správnej normalizácii) dokážeme zachytiť farebnú schému, ktorou pôsobí na čitateľa, presnejšie. Vytvorený obraz je teda na záver prechádzaný a hodnoty farieb jednotlivých jeho pixelov sú vkladané do objektu reprezentujúceho farebnú schému.

Pre úplnú predstavu je priložené porovnanie výstupu pôvodne získaného dokumentu a výstupu po aplikovaní normalizácie obsahu a textových uzlov 3.1.



Obrázok 3.1: Ukážka obrazu originálneho dokumentu, vykresleného vo webovom prehliadači (vľavo), a jeho normalizovanej podoby (vpravo) vykreslenej pomocou CSSBox-u.

Poznámka:

CSSBox v aktuálnej verzii 3.4 nepodporuje vykresľovanie obrázkov na pozadí elementov, preto sa nám nepodarilo zohľadňovať ich vo výslednej farebnej schéme ako to bolo pôvodne navrhované.

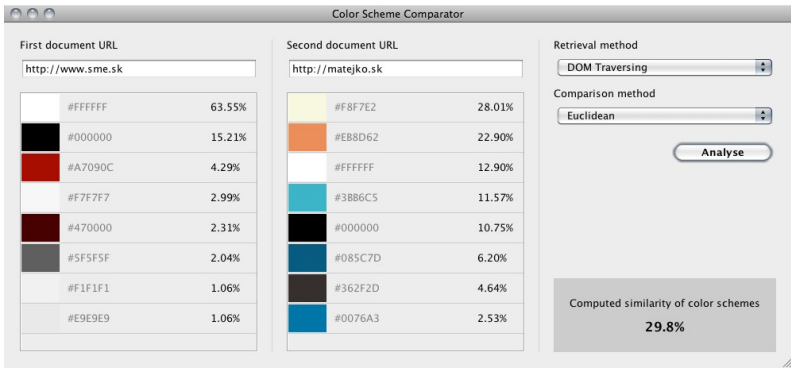
3.4 Porovnávanie farebných schém

V našom riešení je farebná schéma abstrahovaná objektom triedy `ColorScheme`. Tá obsahuje zoznam, v ktorom sú uložené jednotlivé farby vo formáte RGB a informácie o ich výskyte. Objekt `ColorScheme` umožňuje volaním metódy *`getHistogram()`* vytvoriť z uložených farieb histogram, predstaveného objektom triedy *Histogram*. Pri tom sú farby prevádzané do farebného priestoru HSV. Prenášajú sa len farby ktorých výskyt vo farebnej schéme prekročil hranicu minimálneho výskytu 0.5%. Pri pridávaní novej farby do histogramu sú jednotlivé jej zložky zdiskretizované na indexy trojrozmerného poľa. Položka poľa, ktorá sa nachádza výsledným indexom je potom inkrementovaná o zadanú množstvo. Najmenej je v histograme diskretizovaná zložka *hue*, ktorej hodnota sa upravuje do 18 binov aby boli rozlíšené všetky základné rozlíšiteľné farby spektra. Svetlosť má na vnímanú farbu väčší vplyv ako jej sýtosť, preto je kanál *value* diskretizovaný do 9 binov a posledný kanál *saturation*, iba do 5. Takto získaný objekt histogram je už určený na porovnávanie.

Pre samotné porovnávanie implementujeme už predtým priblížené metódy: Euklidovskú metriku a prienik histogramov. Porovnávanie sa realizuje volaním metódy *`getDistance()`* z objektu histogramu. Pri tom sa takisto zadáva, ktorá z dostupných metód porovnávania sa má použiť. Tie sú implementované v triede *Histogram*. Výsledkom porovnávania je interval od 0 do 1 vyjadrujúci percentuálnu podobnosť zadaných porovnávaných histogramov a tak aj podobnosť webových dokumentov z ktorých boli získané.

4 Testovanie

Na účely samotného testovania efektívnosti riešenia, bola vytvorená jednoduchá aplikácia 4.1 využívajúca implementovanú knižnicu. Tá umožňuje v grafickom rozhraní porovnať farebnú schému dvoch dokumentov podľa ich zadaných URL adries. Je pri tom možné vybrať metódu, ktorou budú farebné schémy z dokumentov získané a metódou, ktorou budú medzi sebou porovnané. Po úspešnej analýze dokumentu sú zobrazené rozpoznané farby, ich hexadecimálne číslo a pomerné zastúpenie.



Obrázok 4.1: Grafické rozhranie testovacej aplikácie

4.1 Návrh testovania

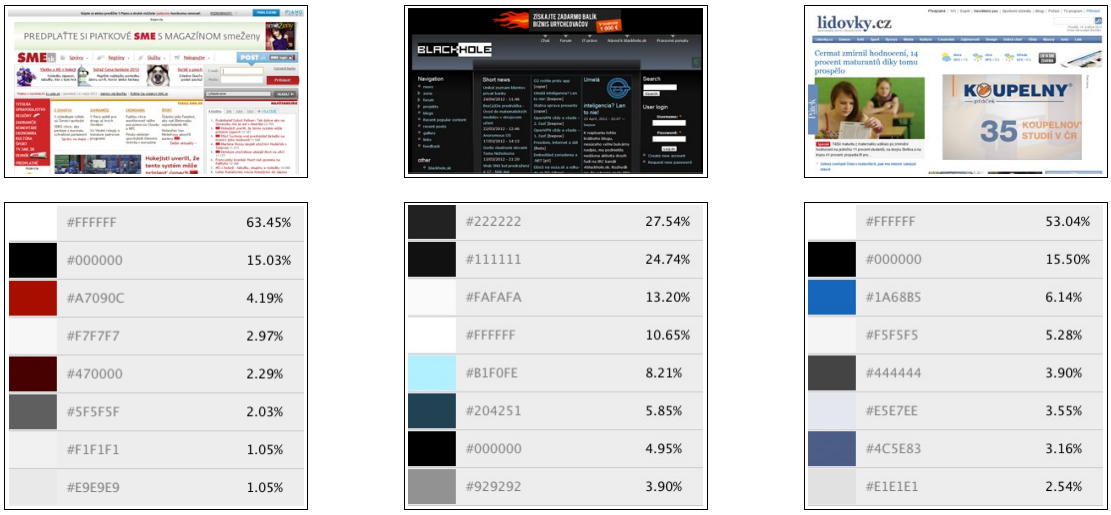
Pomocou tejto aplikácie bola testovaná sada webových stránok, s cieľom zistiť nakoľko implementované metódy zodpovedajú ľudskému vnímaniu ich podobnosti a ktorá z nich pracuje efektívnejšie.

Testovacia sada obsahovala 33 webových stránok, ktoré sa nachádzajú v prílohe práce. Sada bola zostavená tak aby sa v nej nachádzali skupiny podobných stránok – "červených", "modrých" a "čiernych" webov, pričom prvá polovica obsahuje skôr svetlejšie vizuály a koniec sady zas vizuály postavené na čiernom pozadí. Z nich bola vybraná trojica 4.2, ktorá sa následne postupne porovnávali so zvyškom sady.

Podobnosť dokumentov bola najprv vyhodnotená skupinou respondentov, ktorí dvojicu webov označili ako *veľmi podobnú*, *podobnú*, *čiastočne podobnú* alebo *bez podobnosti*. Tieto hranice týchto skupín boli potom na graf určené v úrovniach 85%, 65% a 40%. Porovnávanie ktoré dopadlo pod úrovňou 40% hovorí, že dvojica dokumentov má len minimálnu alebo žiadnu podobnosť, naopak nad hranicou 85% už máme veľmi podobné dokumenty.

Rovnaká úloha bola následne zadaná testovacej aplikácii. Farebná schéma bola z porovnávaných dokumentov zakaždým postupné získaná oboma navrhovanými technikami a porovnaná dvojicou porovnávacích metód.

Každej dvojici porovnávaných dokumentov tak bola pridelených celkom päť hodnotení – štyri automatizované a jedno referenčné od respondentov. Cieľom testovania bolo zistiť nakoľko sa výsledky získané programom približujú k referenčným.



www.sme.sk (14.) *www.blackhole.sk (31.)* *www.lidovky.cz (2.)*

Obrázok 4.2: Webové dokumenty porovnávané s testovacou sadou a ich farebná schéma (získaná metódou priechodu modelom dokumentu)

4.2 Výsledky testovania

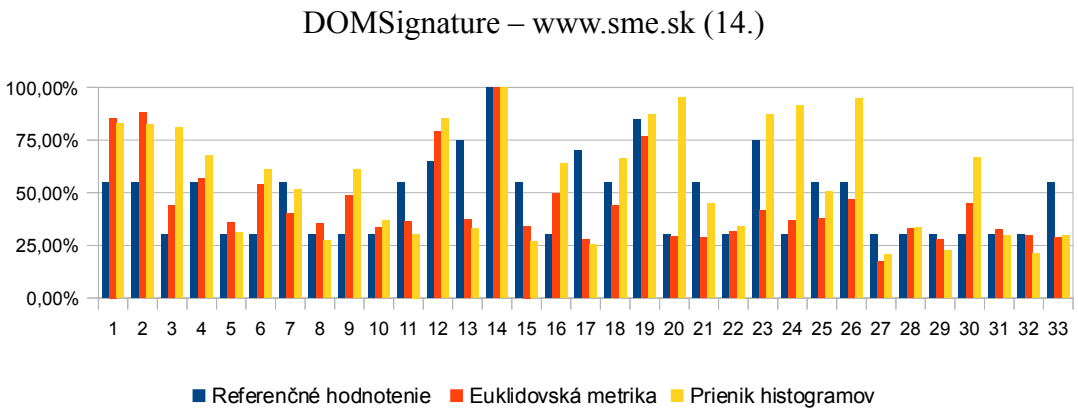
Priebeh testovania je zaznamenaný v grafoch 4.1–4.6. V nich sú na x-ovej osy zaznačené čísla identifikujúce jednotlivé webové dokumenty v testovacej sade a hodnota na y-ovej osy určuje ich percentuálnu podobnosť s práve porovnávaným dokumentom. Táto hodnota je pri každom dokumente uvedená 3-krát – jedna predstavuje podobnosť získanú od respondentov, zvyšné dva ju vypočítavajú sledovanými metódami na porovnávanie histogramov. Pre lepšiu prehľadnosť sú grafy oddelené podľa metódy získavania farebnej schémy (spolu s adresou porovnávanej stránky je toto označenie v nadpise grafu: *DOMSignature* pre metódu priechodu modelom dokumentu a *ViewportSignature* pre metódu analýzy vykresleného obrazu dokumentu).

Za úspešné sme označili porovnávanie, ktorého hodnotenie sa od referenčného odchyľovalo max. do 20% podobnosti. Tolerancia vychádza z rozpätia ktoré sme určili pre hodnotenie respondentov. Ku porovnávaniu každého dokumentu je vypracovaná i tabuľka (4.1–4.3), ktorá vyhodnocuje úspešnosť kombinácie metód. Najlepšia hodnota je zvýraznená.

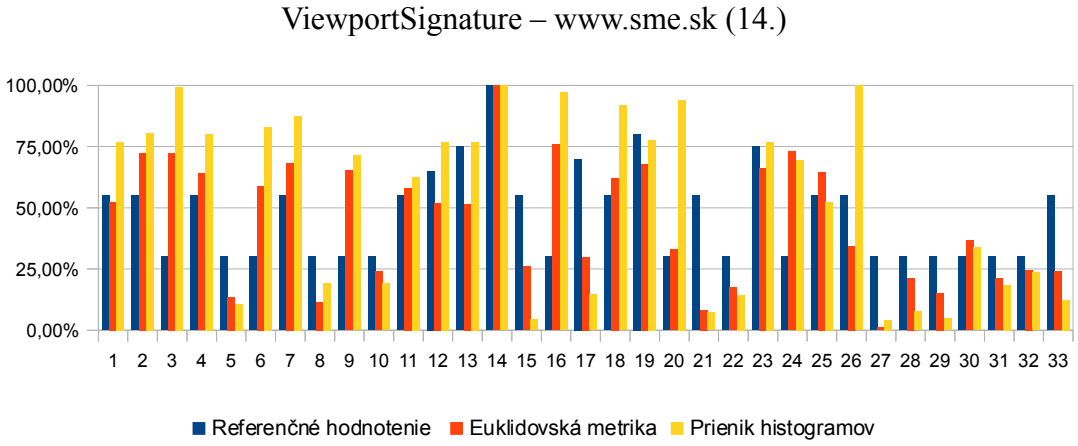
Zo získaných hodnôt sa dá usúdiť, že najlepšie výsledky prináša metóda získavania farebnej schémy pomocou metódy *DOMSignature* a následné porovnávanie prostredníctvom Euklidovskej metriky. Tá dosahuje v priemere 69,7% úspešnosť určovania podobnosti daných webových dokumentov. Metóda *ViewportSignature* však preukazuje o triedu horšie výsledky, a slabšiu efektívnosť – nižšia presnosť a väčšie výpočtové nároky spojené s vytváraním obrazu dokumentu.

Porovnávanie Euklidovskou metrikou vo väčšine prípadov koreluje s referenčnými hodnotami. Naopak, porovnávanie farebných schém pomocou prieniku histogramov neprináša želané chovanie a v testovaní sa táto technika ukázala ako nežiadúca.

Porovnávanie *www.sme.sk* s testovacou sadou



Graf 4.1: Porovnanie získaných podobností farebnej schémy dokumentu www.sme.sk (14.) získanej priechodom modelu dokumentu



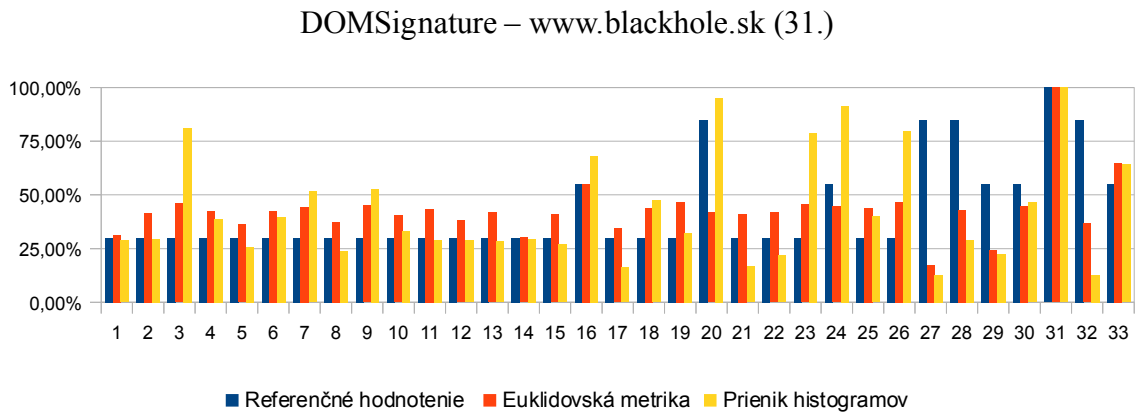
Graf 4.2: Porovnanie získaných podobností farebnej schémy dokumentu www.sme.sk (14.) získanej analýzov vykresleného obrazu dokumentu

	Euklidovská metrika	**Prienik histogramov**
DOMSignature	**69,70%**	57,58%
ViewportSignature	54,55%	30,30%

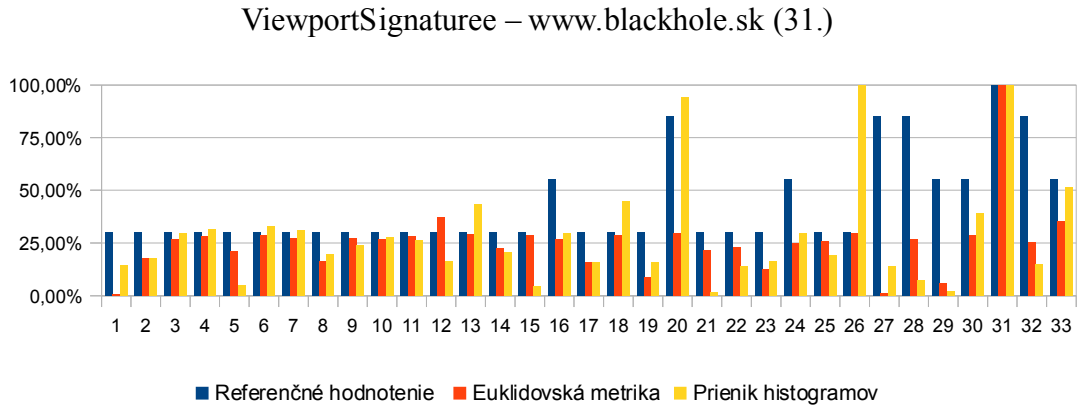
Tabuľka 4.1: Vyhodnotenie úspešnosti porovnávania, pri porovnávaní farebnej schémy dokumentu www.sme.sk

Porovnávaný dokument patril do skupiny "červených" na svetlom podklade. Pomerne dobre však výsledky kopírovali i očakávania respondentov v skupine "modrých" vizuálov a takisto nebola podľa prepokladov nájdená žiadna zhoda v skupine "čiernych" vizuálov.

Porovnávanie *www.blackhole.sk* s testovacou sadou



Graf 4.3: Porovnanie získaných podobností farebnej schémy dokumentu www.blackhole.sk (31.) získanej priechodom modelu dokumentu



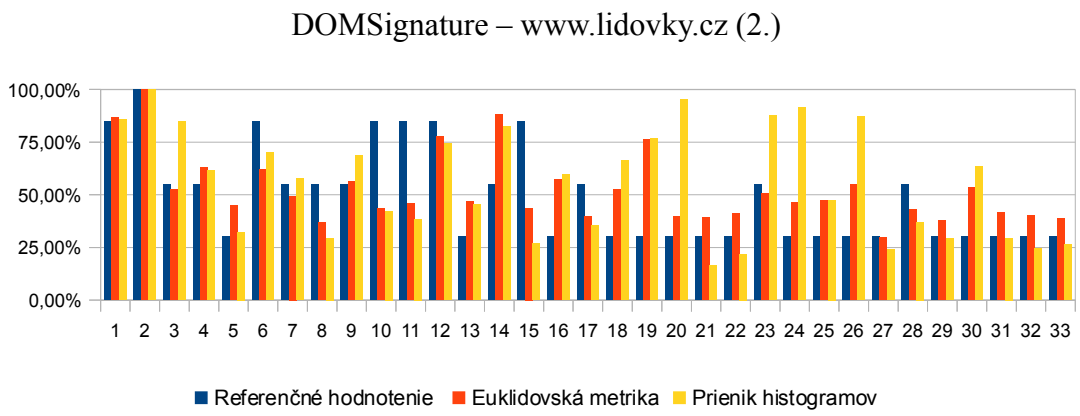
Graf 4.4: Porovnanie získaných podobností farebnej schémy dokumentu www.blackhole.sk (31.) získanej analýzov vykresleného obrazu dokumentu

	Euklidovská metrika	**Prienik histogramov**
DOMSignature	**63,64%**	42,42%
ViewportSignature	39,39%	39,39%

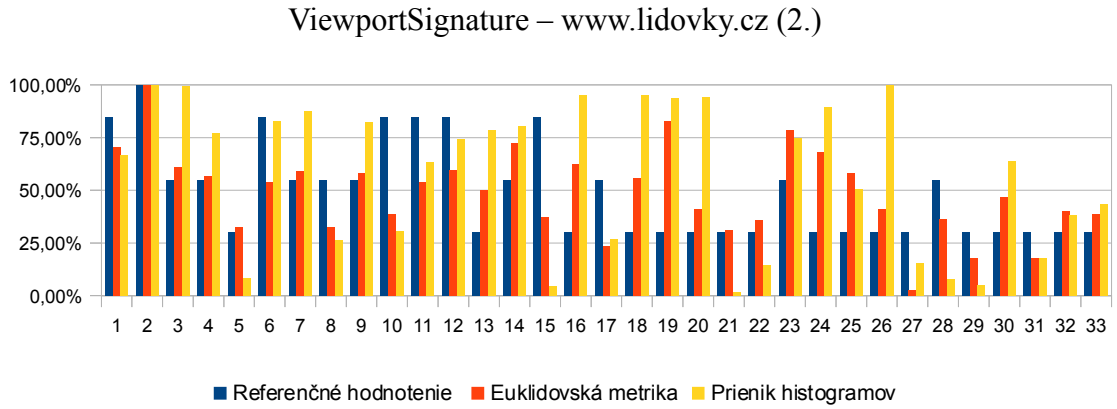
Tabuľka 4.2: Vyhodnotenie úspešnosti porovnávania, pri porovnávaní farebnej schémy dokumentu www.blackhole.sk

Porovnávaná stránka v tomto prípade parila do skupiny "čiernych". Podľa očakávaní nebola pri vizuáloch na svetlom podklade nájdené žiadne zhody, avšak trochu prekvapujúce chovanie bolo spozorované pri porovnávaní vo vlastnej skupine kde sa podarilo priblížiť len dvom výsledkom očakávaným od respondentov. Obzvlášt pri stránkach 27 a 29 pri ktorých sa ani jednej z metód nedarilo získať relevantnú farebnú schému.

Porovnávanie *www.lidovky.cz* s testovacou sadou



Graf 4.5: Porovnanie získaných podobností farebnej schémy dokumentu www.lidovky.cz (2.) získanej priechodom modelu dokumentu



Graf 4.6: orovnanie získaných podobností farebnej schémy dokumentu www.lidovky.cz (2.) získanej analýzov vykresleného obrazu dokumentu

	Euklidovská metrika	**Prienik histogramov**
DOMSignature	**75,76%**	45,45%
ViewportSignature	42,42%	33,33%

Tabuľka 4.3: Vyhodnotenie úspešnosti porovnávania, pri porovnávaní farebnej schémy dokumentu www.lidovky.cz

Posledná stránka patrila do skupiny "modrých" na svetlom podklade. Jej podobnosť bola metódami vyhodnotená najlepšie v celom teste. Hlavne Euklidovská metrika verne kopírovala očakávanú podobnosť, okrem dokuemntov 10–12, ktoré užívatelia vysoko hodnotili aj napriek tomu, že ich svetlosťou až tak nezodpovedá porovnávanému dokumentu. Na metódu sú ale kladené očakávania že dokáže toto vnímanie zohľadniť.

Záver

Výsledky testovania s implementovanou knižnicou ukazujú, že je to použiteľný nástroj, aj keď nedosahuje až také kvality ako iné formy porovnávania webových dokumentov prostredníctvom ostatných vizuálnych rysov, resp. ich kombináciou.

Ako možné rozšírenie vidím skvalitnenie normalizácie obsahu, aby zahrňovala aj ďalšie používané konštrukcie štruktúry dokumentu. Taktiež by výsledky spresnilo, ak by do farebnej schémy boli zahrnuté aj obrázky používané na pozadí elementov, ako som uviedol v pôvodnom návrhu. Množstvo súčasných webov rieši nasadenie grafiky, ktorá zásadne ovplyvní výsledný dojem z vizuálu, práve prostredníctvom nich. Podobne značnú časť grafiky si už autori zvykajú realizovať prostredníctvom CSS gradientov.

Širším problémom pri získavaní farebnej schémy je aj množstvo technológií ktorými dnes môže byť vizuál dokumentu riešený – od už zaužívaného FLASHu až po CSS animácie a použitie WebGL. Tie robia web skutočne dynamickým, avšak komplikujú úlohu zachytiť farebnú schému ako pri tradičnej statickej stránke.

Podobne ako v [9] je potom možné dospieť k úvahám, že porovnávanie prostredníctvom farebnej schémy dokumentu síce je realizovateľné, ale spoľahlivé použitie v bežnom prostredí webu nie príliš praktické. Uplatnenie si však stále môže nájsť pri porovnávaní v sade dokumentov s vymedzenými vizuálnymi prvkami, napr. šablóny rozšírených blogov a redakčných systémov, ktoré spravujú rôzne template služby. Prínosom by mohlo byť takisto aj jej nasadenie spolu s ďalšími metódami pre porovnávanie vizuálnych rysov dokumentov, čím by vzniklo komplexné riešenie.

Literatúra

[1] CSSBox [online], <http://cssbox.sourceforge.net/>

[2] Schafer, S. M.: *HTML, XHTML, and CSS Bible*. John Wiley & Sons, 2011, ISBN 1118081307

[3] The global structure of an HTML document [online]. [cit. 21.2.2012], <http://www.w3.org/TR/html401/struct/global.html>

[4] Cascading Style Sheets [online]. [cit. 21.2.2012], <http://www.w3.org/Style/CSS/Overview.en.html>

[5] CSS – Syntax and basic data types [online]. [cit. 21.2.2012], <http://www.w3.org/TR/CSS21/syndata.html>

[6] Lopresti, D., Wilfong, G.: *Comparing Semi-Structured Documents via Graph Probing* [online]. 2002. [cit. 16.12.2011]. Dostupný z WWW: <http://www.mis2003.unina.it/MIS2001/first/dan.pdf>

[7] Alpuente, M., Romero, D.: *A Visual Technique for Web PagesComparison* [online]. 2008 [cit. 16.12.2011].

[8] Rishav Chakravarti, Xiannong Meng: *A Study of Color Histogram Based Image Retrieval* [online]. 2009 [cit. 22.3.2012]. Dostupný z WWW: <http://aimm02.cse.ttu.edu.tw/class_2009_2/CV/OpenCV/References/A%20Study%20of%20Color%20Histogram%20Based%20Image%20Retrieval.pdf>

[9] Kudělka, M., Takama, Y., Snášel, V., Klos, K., Pokorný, J.: *Visual Similarity of Web Page* [online]. 2009 [cit. 22.3.2012].

[10] Lowe, D. G.: *Distinctive Image Features from Scale-Invariant Keypoints* [online]. 2004 [cit. 16.12.2011].

[11] Bay, H., Ess, A., Tuytelaars, T., Van Gools, L.: *Speeded-Up Robust Features (SURF)* [online]. 2006 [cit. 16.12.2011]. Dostupný z WWW: <ftp://ftp.vision.ee.ethz.ch/publications/articles/eth_biwi_00517.pdf>

[12] Žára, J., Beneš, B., Sochor., J., Felkel., P.: *Moderní počítačová grafika*. Computer Press, 2004, ISBN 80-251-0454-0

[13] Color Models [online]. <http://learn.colorotate.org/color-models.html>

[14] Jain, A. K., Vailaya, A.: *Image Retrieval using Color and Shape* [online]. 15.5.1995 [cit. 11.4.2012]. Dostupný z WWW: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.47.9699>

[15] Jeong, S.: *Histogram-Based Color Image Retrieval* [online]. 15.3.2001 [cit. 8.12.2011]. Dostupný z WWW: <scien.stanford.edu/pages/labsite/2002/psych221/projects/02/sojeong/#dct>

[16] Betancourt, M.: *A Bayesian Approach To Histogram Comparison* [online]. 28.9.2010 [cit. 20.10.2011]. Dostupný z WWW: <http://web.mit.edu/~betan/www/papers/bayesHistComp.pdf>

[17] NekoHTML [online], <http://nekohtml.sourceforge.net/>

[18] org.w3c.dom [online], <http://docs.oracle.com/javase/1.4.2/docs/api/org/w3c/dom/package-summary.html>

[18] JStyleParser [online], <http://cssbox.sourceforge.net/jstyleparser/>

Zoznam príloh

Príloha č. 1

Testovacia sada webových dokumentov

Zoznam webových dokumentov obsianutých v sade, zvyŕaznené sú farebné skupiny, do ktorých boli jednotlivé dokumenty zaradené ("modré", "červené" a "čierne"):

1. *http://en.wikipedia.org/wiki/Color_histogram*
2. *http://www.lidovky.cz/*
3. *http://www.fit.vutbr.cz/*
4. *http://chmu.cz/*
5. *http://www.superwebhosting.sk/*
6. *http://www.facebook.com/*
7. *http://www.nrsr.sk/web/*
8. *http://www.gametrailers.com/*
9. *http://muni.cz/*
10. *http://ejohn.org/*
11. *http://www.shmu.sk/*
12. *http://www.ta3.com/*
13. *http://www.mbank.sk/*
14. *http://www.sme.sk/*
15. *http://www.henkel.sk/*
16. *http://www.opera.com/*
17. *http://www.zitbrno.cz/*
18. *http://www.theredpages.co.uk/*
19. *http://edition.cnn.com/*
20. *http://www.sonze.com/*
21. *http://www.tubatomic.com/revolver/*
22. *http://darkcrimson.com/*
23. *http://jakeprzespo.com/*
24. *http://www.rob-james.com/*
25. *http://dougdosberg.com/*
26. *http://dakshadesign.com/*
27. *http://blackestate.co.nz/*
28. *http://jquery.com/*
29. *http://www.supermusic.sk/*
30. *http://rockforpeople.cz/*
31. *http://blackhole.sk/*
32. *http://www.realitymod.com/*
33. *http://www.chromebagsstore.com/*

Zoznam dokumentov ktoré sa v práci porovnávali s celou testovacou sadou:

2. *http://www.lidovky.cz*
14. *http://www.sme.sk*
31. *http://www.blackhole.sk*

Príloha č. 2

Návod na použitie

Návod sa vzťahuje na použitie knižnice *ColorSchemeComparator* prezentovanej v práci.

Postup spustenia demo aplikácie prezentujúcej funkcionalitu knižnice:

- Demo aplikáciu spustíte v adresári /ColorSchemeComparator príkazom
  `java -jar  ColorSchemeComparator.jar`

Postup nasadenia knižnice do vlastného projektu:

- Skopírujte adresár `/ColorSchemeComparator` projektu na Vami zvolené umiestnenie
- Naimportuje nakopírovaný adresár projektu do Vášho projektu.

Podrbná dokumentácia súčastí knižnice *ColorSchemeComparator* sa nachádza v adresári `/ColorSchemeComparator/doc`.